

STRING TİPİ

▼ String türü ve string gerçeği

- Metinsel ifadeleri tutabilen özel bir türdür.
- String referans türü bir değişkendir. Referans türü olup da programlamada bir keyword ile karşılanabilen tek tür string türüdür.
- Stringin değeri HEAP'te tutulur.
- String ifadeler esasında bir char dizisidir. Yazılımda string ifade yoktur. Aslında karakterlerin bir araya gelmiş hali vardır.
- String özünde bir char dizisi olabilir ama yapısal olarak yine de string bir ifade olduğu için referanslı atılmaz, Array ile karşılanamaz.

```
#region String - char Dizisi
//string ifadeler esasında bir char dizisidir. Yani yazılım açısından string ifade yoktur! Esasında karakterlerin bir araya gelmiş hali vardır.
//Dolayısıyla karakterleri bir araya getirebilecek yegane şey bir dizidir. String ifadeler yazılımsal açıdan bilgisayarda bir char dizisi olarak tarif
//edilmekte ve o şekilde tutulmaktadır.
//String ifadeler özünde bir char dizisi/yani dizi olmasından dolayı referans türü değişkenlerdir. Çünkü diziler referans türüdürler. yani
//nesnelerdir.yani heapte tutulurlar.

string metin = "sebepsiz boş yere ayrılacaksan...";
//string ifadeler char dizisi olduklarından dolayı yapısal olarak her bir karakter baştan sona otomatik indexlenmektedir. Dolayısıyla string bir ifade
//üzerinde bizler indexer operatöründe kullanabilmekteyiz...

Console.WriteLine(metin[3]);
Console.WriteLine(metin.Length);

//Array array = metin; //String özünde bir char dizisi olabilir ama velakin yapısal olarak yine de string olduğu için Array referansına atılamaz, Array
//ile karşılanamaz!!!
#endregion
```

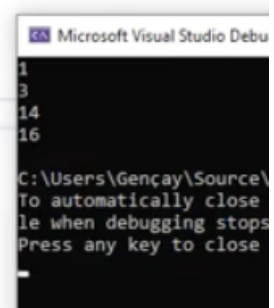
```
#region String - char Dizisi

string metin = "sebepsiz boş yere ayrılacaksan...";

for (int i = 0; i < metin.Length; i++)
{
    if (metin[i] == 'e')
        Console.WriteLine(i);
}

I

#endregion
```



-

▼ Null ve empty durumları ve farkları

▼ Null

- Bir değilken/ nullable / referans eğer ki null alıyorsa bellekte bir karşılığı yok demektir.Arsa yok
- Null alabilen türler referans türlerdir.
- Değer türlü değişkenlerin null alabilmesi için nullable olmaları gerekmektedir.
- Nullable nedir peki?
 - Değer türlü değişkenin null alabilmesini sağlamak için nullable hale getirmemiz gerekir. Bunu da şu şekilde yaparız.
 - `int? a = null;` şeklinde
- Null olan bir değer üzerinde işlem yapmaya çalıştığımızda runtime hatası alırız.

▼ Empty

- Alan tahsisi yapılıyor ama içinde değer yok anlamına gelir. Arsa var ama içinde ev yok gibi düşünebilirsin.
- Tüm değerlere empty atanabilir.
- `int a = 0; bool b = false; int[] x = new int [55];` → diziyi tanımladık ama içi boş.
- Empty denildiğinde aklımıza string'e " " değerinin verildiği gelsin.!!
 - `string a = "";`
 - `string a2 = string.Empty;` kullanılabilir.
- Empty olan bir değer üzerinde işlem yapabiliriz.

▼ IsNullOrEmpty

- Elimizdeki string ifade null mü yoksa empty mi kontrol etmeliyiz.
- Normalde şu şekilde kontrol edebiliriz.

```
//if (x != "")
//if (x != string.Empty && x != null)
if (x != string.Empty && x is not null)
{
    //Operasyon...
}
```

- Fakat IsNullOrEmpty ile kontrol ettiğimizde bu fonksiyon check yapıp bize boolean tür döndürür.

```
//Eğer ki değer null ya da empty ise geriye true değilse false dönecektir.
if (!string.IsNullOrEmpty(x))
{
    //Operasyon...
}
```

•

▼ IsNullOrWhiteSpace

- Fonksiyon elimizdeki string ifadenin null, empty, ya da boşluk karakterlerinden ibaret olma durumunu kontrol eder.

▼ String'in RAM (Heap) ilişkisini inceleyelim

- String ifadelerin değerleri referans türü olduklarından dolayı bir nesnedir. Nesne dediğimizde heap'de tutulur.
- String değişkenler referans olduklarından dolayı stack de tutulacaklardır.

▼ String ifadelerde + operatörü

- İki string ifade arasında birleştirme görevi görür.
- Birleştirme görevi görür.

```
using System;

static void Main(string[] args)
{
    #region String ifadelerde + Operatörü
    //String ifadelerde + operatörü kullanılabilir.
    //İki string ifade arasında birleştirme görevi görür.
    string a = "merhaba", b = "dünya";
    Console.WriteLine(a + b);
    //Bir string ifade ile herhangi bir tür + operatörüyle işleme tabi tutulabilir.
    //+ operatörü string bir ifadeyle herhangi bir türdeki ifadeleri işleme tabi tutarken object + string olarak davranış sergileyecek ve sonuç olarak geriye string değer döndürecektir.
    //Dolayısıyla herhangi bir ifadeyi string'e dönüştürebiliriz.

    //int a2 = 5;

    //a + a2

    Console.WriteLine(5 + 7 + 20 + "ahmet");

    #endregion
}
```

Microsoft Visual Studio Debug Console

```
merhabadunya
32ahmet

C:\Users\Gencay\Source\Repos\Application\string_turu\bin\Debug\net5.0\string_turu.exe (process 4736) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

- object+ string

▼ String formatlandırma

- Elimizdeki değerleri uygun yerlere yerleştirmemizi sağlayan programatik bir rapordur.
- ### ▼ + operatörü ile (Çok tercihimiz diil)

```
space string_turu
// references
class Program
{
    // references
    static void Main(string[] args)
    {
        #region String Formatlandırma
        string isim = "Gençay", soyisim = "Yıldız", tcNo = "12345678910";
        int yas = 28;
        bool medeniHal = true;

        Console.WriteLine("TC No : ..... olan ..... şahsın bilgileri | Yas : .. | Medeni Hal : .. ");

        #region + Operatörü
        Console.WriteLine("TC No : " + tcNo + " olan " + isim + " " + soyisim + " şahsın bilgileri | Yas : " + yas + " | Medeni Hal : " + (medeniHal ? "Evli" : "Bekar"));
        #endregion
        #endregion
    }
}
```

+ ile string formatlandırmada ternary operatörü kullanılıyorsa bunu parantez içersine almanız gerekmektedir...

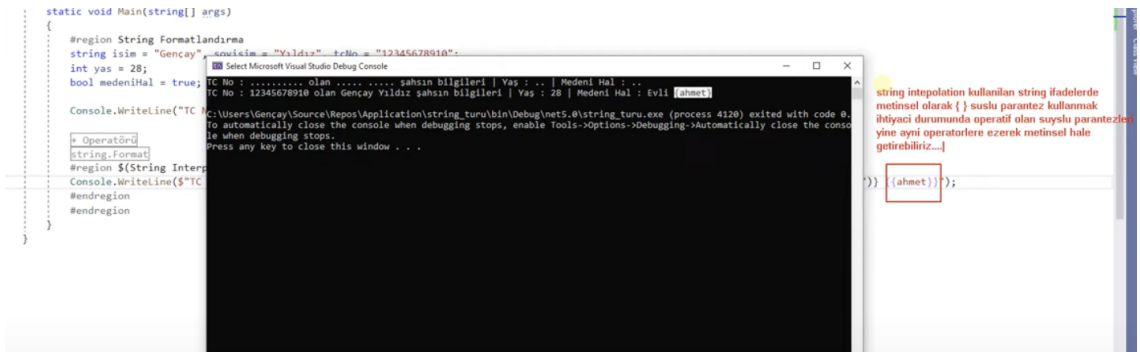
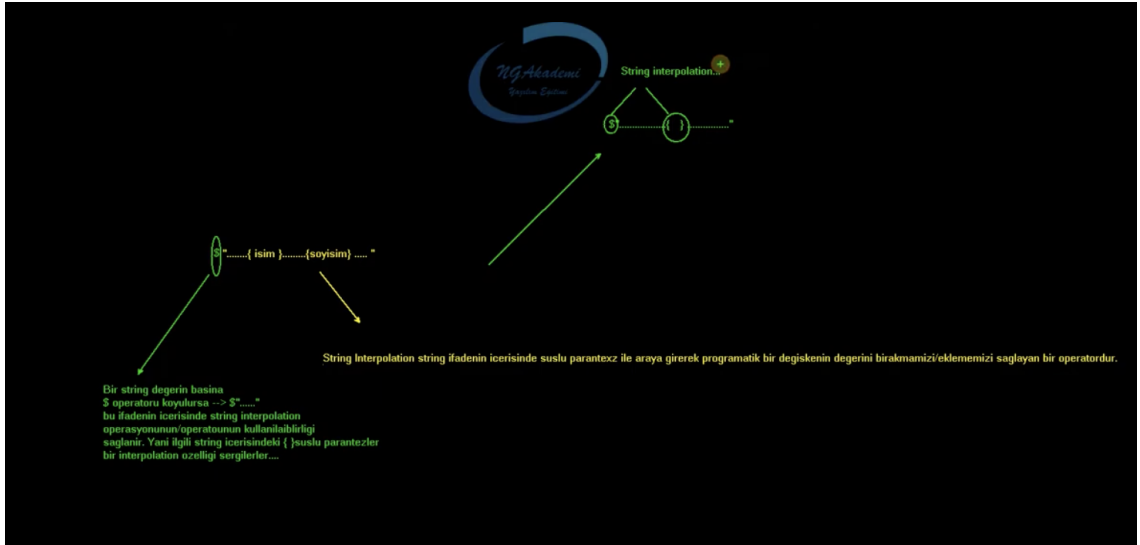
▼ String.Format metodu ile

```
// references
static void Main(string[] args)
{
    #region String Formatlandırma
    string isim = "Gençay", soyisim = "Yıldız", tcNo = "12345678910";
    int yas = 28;
    bool medeniHal = true;

    Console.WriteLine("TC No : ..... olan ..... şahsın bilgileri | Yas : .. | Medeni Hal : .. ");

    #region String.Format
    string sonuc = string.Format("TC No : {0} olan {1} {2} şahsın bilgileri | Yas : {3} | Medeni Hal : {4} ", tcNo, isim, soyisim, yas, medeniHal ? "Evli" : "Bekar");
    Console.WriteLine(sonuc);
    #endregion
    #endregion
}
```

▼ String Interpolation(\$) Operatörü ile



▼ String kaçış karakterleri (Escape)

- \ sola yatan slaşşşş kullanılır.
- Kendisinden sonra gelen ifadenin eylemsel olmadığını gösterir.
- \yanına illa ki özel bir karakter bekler. Aksi takdirde hata verir. Metinsel olarak kullanmak istiyorsak eğer ilgili karakteri yine kendisiyle ezmeliyiz. \\ şeklinde.
- Bu karakteri kullanarak mesela string ifade içinde metinsel bir tırnak işareti kullanmak durumunda kaldığımda \" şeklinde kullanabilirim.

```

6 | {
7 |     0 references
8 |     static void Main(string[] args)
9 |     {
10 |         #region String Kaçış Karakterleri
11 |         #endregion
12 |     }
13 | }
14 |

```

Eğer ki bu şekilde string için özel eylemsel mahiyet ifade eden bir karakteri metinsel olarak kullanacaksam bu karakterin o anlık özel karakter olmadığını ifade etmem gerekmektedir.

Bunun için escape/kaçış karakterleri kullanılmalıdır.

String içerisinde kaçış karakteri olarak \ (Backward Slash) kullanılmaktadır. String içerisinde özel/operatif karakterleri ezen ve bunları metinsel hale getirmemizi sağlayan bir karakterdir.

Eylemsel bir karakter. String açısından belirli bir operasyonu/eylemi/sorumluluğu üstlenen bir karakterdir.

Dolayısıyla böyle bir karakteri metnin içerisinde salt bir şekilde kullanmamız mümkündür.

Kaçış Karakteri	Karakterin Açıklaması
\0	Null sonlandırma karakteridir. Genel olarak dosya veya veri kanalının bitişini belirtmek için kullanılır.
\a	Bip sesini çıkartan karakterdir
\b	BackSpace – Geri – Önceki karakteri silme
\t	Tab
\r	Satır başı (Carriage Return)
\n	Bir alt satıra iner
\v	Dikey Tab
\f	Sayfa ilerleme
\"	Çift tırnak
\'	Tek tırnak
\\	Backslash

▼ @(Verbatim String) Operatörü

▼ ilk kullanım

- Bir değişken metot vs gibi yapıların programatik bir keyword'e karşılık gelmesi mümkün değildir. Derleyici hatası verilir.
- `int @void =5;` gibi..

▼ ikinci kullanım

- Escape operatörünün yapması gereken şeyleri de yapabiliyor örneğin;

```
string metin = @"hava çok ""güzel"" ";
```

Burada verbatim operatörü kullandıktan sonra çift tırnağı çift tırnakla ezebiliyoruz. Bunun çıktısı hava çok "güzel" olacaktır.

- `int @void =5;` gibi..

▼ Üçüncü kullanım

- Üstteki kullanıma izin vermez derleyici:

```

2. Kullanım
#region 3. Kullanım
string metin = "masldmkasm
               dkasmdkm
               aksmdkasmdkmaskdm";

string metin = "kamsdkamsdkmasd" +
               "kamsdkmaskdmasd" +
               "alksmödläösmd" +
               "asldkmaksdm" +
               "asdasd";

```

Üstteki kullanımın başına @ koyarsak, bunu algılayabilir ve kabul eder derleyici



▼ String Interpolation ve verbatim string birlikteliği

```

string mailMessage = @$"Merhaba ..... \n
... nolu sipariş talebiniz tarafımızca alınmıştır.\n
Fiyat : .... ";
#endregion

```

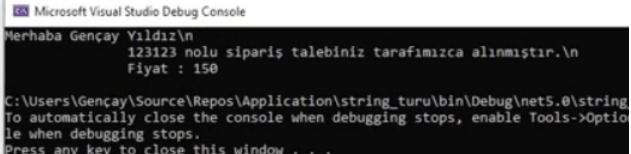
@ ope. ile \$ ope. birlikte kullanılıyorsa önce @, sonra \$ bildirilmelidir.

- Bunun çıktısı şöyle olur:

```

string mailMessage = @$"Merhaba {isim} {soyisim}\n
{siparisNo} nolu sipariş talebiniz tarafımızca alınmıştır.\n
Fiyat : {fiyat} ";
Console.WriteLine(mailMessage);
#endregion

```



- Aslında bunu şöyle yapmalıyız:

```
string mailMessage =  
@"Merhaba {isim} {soyisim}  
{siparisNo} nolu sipariş talebiniz tarafımızca alınmıştır.  
Fiyat : {fiyat} ";  
Console.WriteLine(mailMessage);  
#endregion  
}  
}
```

Microsoft Visual Studio Debug Console

```
Merhaba Gençay Yıldız  
123123 nolu sipariş talebiniz tarafımızca alınmıştır.  
Fiyat : 150  
  
C:\Users\Gençay\Source\Repos\Application\string_turu\bin\Deb  
To automatically close the console when debugging stops, ena  
le when debugging stops.  
Press any key to close this window . . .
```